GPS-free Disaster-Scale Mapping and Energy-Efficient Alerting Scheme in a Wireless Sensor Network

Ehssan Sakhaee, Naoki Wakamiya, Masayuki Murata
Graduate School of Information Science and Technology
Osaka University
{sakhaee, wakamiya, murata}@ist.osaka-u.ac.jp

Abstract

In this paper, we present a disaster-scale mapping and energy-efficient alerting scheme that can be used in disaster monitoring environments that are prone to node failure and destruction due to the occurrence of disasters, and are generally deprived of GPS capabilities and or availability. The mapping scheme is able to provide a distributed mean of monitoring and reporting disaster's location which is triggered by failure of sensor nodes due to their inevitable destruction. Upon reporting, not only the location of the disaster, but also the disaster's scale with regard to the approximate area size, is calculated at the sink using the information received via nodes that detect their failed neighbors. The mapping scheme is designed so that the sink is able to obtain the location of sensor nodes simply from their unique duple-IDs.

1. Introduction

Recently, environmental sensor networks have become an important research focus, due mainly to the increasing concern of our environment, our impact on it, and the impact it has on us and our future. The monitoring and reporting of environmental disasters is a critical issue for their effective management and treatment, particularly in remote areas. Application of this can be realized in many facets. In particular the reporting of disaster locations and their scale such as explosions, fires and other various destructions caused to the environment has not been treated in previous literature, although it should be an important application area of sensor networks. Furthermore, such scenarios may have certain limitations and characteristics, such as limited or no availability of global positioning system (GPS), either due the high cost and high energy consumption of GPS devices, making them unsuitable for sensor nodes, especially in an environment where sensor node destruction is inevitable. Furthermore there are environments where GPS reception is deterred, e.g. in enclosures, mines, underwater, underground etc. Furthermore, the scale of the disaster in terms of its geographical size is crucial in order to realize the seriousness of the disaster, and to identify the correct course of action. For this reason, this paper aims to address such issues in the proposed protocols. The primary aims and assumptions of the proposed schemes are as follows: 1) a sensor network which reports disaster scale (in terms of area/parameter size). 2) Destruction of sensor nodes in the disaster area is presumed. 3) Only nodes which physically sense the disaster and neighbor nodes leading *in the direction* of the sink take part in routing *alerts*. This effectively saves the energy of other nodes which do not lead towards the sink in the rest of network, and prevents flooding of entire sensor network, reducing the total energy consumption of the network. 4) It is assumed that the nodes are static in this network and their placement follows rules outlined in the next section. The basic approach of the approach consists of the following: 1) When a disaster occurs, the nodes in the affected region are destroyed. 2) The nodes surrounding the affected area notice the disaster by death of neighbor nodes (residing in the affected area). 3) Such neighbors issue an alert message, which is forwarded to the sink by intermediate nodes.

Traditional localization schemes, such as those surveyed in [1] assume the existence of nodes that know their own location (known as seeds) from which other nodes estimate their positions. This however imposes a limitation on environments where the destruction of nodes is inevitable, such as disaster areas, and where disaster monitoring is required. Hence, the destruction of seeds in a disaster will cause the scheme to fail with regards to localization of other nodes as they would then have no reference point. Our scheme does not assume the existence of seeds, and is purely distributed. Nodes simply know their relative position to the sink from their own location identifiers and do not rely on neighboring nodes to discover their relative positions. Furthermore, nodes are able to find new paths to sink, in case nodes on the current path fail.

978-0-7695-3330-8/08 $25.00 © 2008 IEEE
DOI 10.1109/SENSORCOMM.2008.30

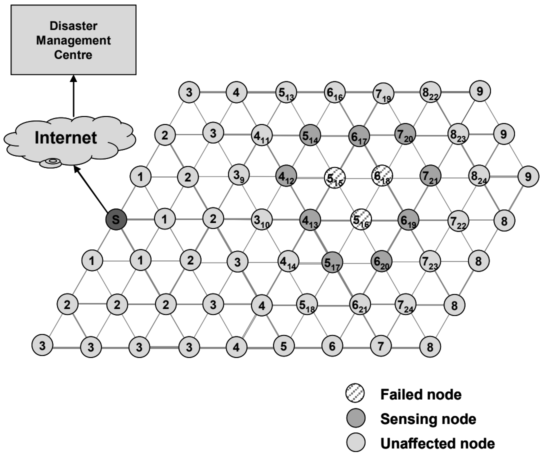

Figure 1. Example scenario of a disaster in HexNet.

The mapping scheme dubbed *HexNet* allows nodes to report disasters and for the position and scale (size) of disaster to be evaluated at the server/user end.

2. Background

A vast number of methods are available for the purpose of localization of an ad hoc or sensor network, many of which have been surveyed in [1]. The most obvious may be the use of global positioning system (GPS) by equipping each node with a GPS reception, allowing every node to obtain its position directly via the GPS satellite. However, the cost of GPS receivers is high and not suitable for use in sensor networks, especially in environments where sensor networks are prone to destruction. Furthermore, GPS receivers consume large amount of energy [1], again a major issue in sensor networks with limited energy source and battery life.

There are numerous localization schemes which use various means to estimate the location of nodes. Some of these utilize the hardware of sensor nodes (range-based localization), such as those used in [2], [3] and [4] which use the received signal strength indication (RSSI) to compute and estimate ranges. However, these approaches may render inaccuracies depending on environmental conditions [1] and hence may not be suitable for environmental sensor networks. Other hardware ranging schemes include the use of time of arrival and time difference of arrival (TDoA) as in [5], [6] and [7]. In [8] the Angle of Arrival of signals is used. Additionally there are range-free solutions which do not require the physical properties mentioned so far. In [9] and [10] a gradient method called multilateration

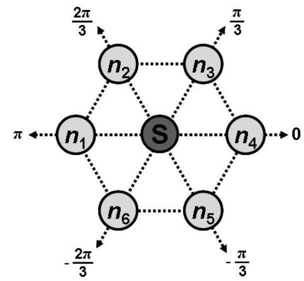

Figure 2. Angular IDs for the *n*th Range ID nodes.

is used to compute ranges. In [11] a distributed algorithm is used to estimate location of nodes using Bezier curves. However, both the range-based solutions and range-free solutions need the existence of "seeds", nodes which possess their location at all times, in order for the other nodes to effectively estimate their own locations. Limitation is imposed in environments where the destruction of nodes is inevitable, such as environments where sensors are used for the purpose of monitoring environmental disasters. In addition to localization schemes, there exist "location-aware" routing protocols which provide scalable and energy-efficient routing in wireless sensor networks [12-14]. However, these protocols presume that the nodes already know their location positions, for example using a location server or some other means, such as the use of GPS. The use of sensor networks for the purpose of environmental monitoring and reporting is not new. Applications of sensor networks can include, but not limited to habitat monitoring, changes in environmental conditions affecting crops and livestock, planetary exploration, and chemical/biological detection [15-24].

3. HexNet mapping scheme

3.1. Overview of HexNet

HexNet is a mapping scheme used for sensor networks in order to effectively retrieve the location and scale (geographical size) of a disaster. A layout of such a network is shown in Fig. 1. In this scenario, several nodes have failed (destroyed), and their failure is detected by the nodes surrounding the area, which are termed as *sensing nodes* in this paper. The sensing nodes then immediately report alert messages back to the sink. The sink is then able to retrieve the location and the size of the disaster from the information obtained from the alert messages of individual sensing nodes.

3.2. HexNet mapping configuration

The HexNet mapping scheme proposed for the purpose of reporting Disaster-Scale information consists of sensor nodes uniformly distributed across the terrain in which monitoring is required. The outline of the mapping scheme is as follows: 1) Sensors are arranged at equal distances from each other in a hexagonal-style grid as shown in Fig. 1, dividing the sensor nodes around hexagonal regions. 2) Each node has *identifiers* that reflect their *relative location* with regard to the *sink* (labeled "S" in Fig. 1). Each node's unique identifier is a combination of two separate identifiers involving a Range ID (RID) and Angular ID (AID) in the form of RID_{AID}. Although neither the RID nor AID is unique by itself, the combination of the two is unique for each node in the network. 3) Sensors closer to the sink have a lower Range ID than nodes further away. The sensors' Range IDs increment with increasing distance from sink (node S), as shown in Fig. 1. 4) The AID increments for each repeating RID in clockwise manner, starting at a value of 1 at π radians and incrementing every π/3 radians as shown in Fig. 2. The AID is given by

$$AID = 3n(1 - \theta/\pi) + 1 \qquad (1)$$

where n is node X's RID and θ is the relative angle of node X from the sink.

3.3. Position and distance of nodes

The topology used for HexNet is established in such a way that the relative position and distance of nodes from the sink can be calculated from their unique RID and AID combination. Using the sine rule, the actual physical distance D from the sink to node X is

$$D = \frac{nd\sqrt{3}}{2\sin(\theta + \pi/3)} \qquad (2)$$

where d is the distance between each node in the network (constant), n is the current node's RID, and θ is the relative angle of node X from the sink given by

$$\theta = \frac{\pi}{3n}(3n - AID + 1) \qquad (3)$$

Hence the Cartesian position of node X is ($D\cos\theta$, $D\sin\theta$) assuming the sink is taken to be located at the origin with a Cartesian position of (0, 0).

The above conditions ensure that the position of any nodes can be determined by the sink from the combination of the duple-IDs RID and AID.

Table 1. Neighbor node's responses to the alive-request messages.

Sensing Node	Expected Replies	Actual Replies
5_{14}	$4_{11}5_{13}6_{16}6_{17}5_{15}4_{12}$ (6)	$4_{11}5_{13}6_{16}6_{17}4_{12}$ (5)
6_{17}	$5_{14}6_{16}7_{19}7_{20}6_{18}5_{15}$ (6)	$5_{14}6_{16}7_{19}7_{20}$ (4)
7_{20}	$6_{17}7_{19}8_{22}8_{23}7_{21}6_{18}$ (6)	$6_{17}7_{19}8_{22}8_{23}7_{21}$ (5)
7_{21}	$6_{18}7_{20}8_{23}8_{24}7_{22}6_{19}$ (6)	$7_{20}8_{23}8_{24}7_{22}6_{19}$ (5)
6_{19}	$5_{16}6_{18}7_{21}7_{22}7_{23}6_{20}$ (6)	$6_{18}7_{21}7_{22}7_{23}6_{20}$ (5)
6_{20}	5_{17} $5_{16}6_{19}7_{23}7_{24}6_{21}$ (6)	$5_{17}6_{19}7_{23}7_{24}6_{21}$ (5)
5_{17}	$4_{14}4_{13}$ $5_{16}6_{20}6_{21}5_{18}$ (6)	$4_{14}4_{13}6_{20}6_{21}5_{18}$ (5)
4_{13}	$3_{10}4_{12}5_{15}5_{16}5_{17}4_{14}$ (6)	$3_{10}4_{12}$ $5_{17}4_{14}$ (4)
4_{12}	$3_{9}4_{11}5_{14}5_{15}4_{13}3_{10}$ (6)	$3_{9}4_{11}$ $5_{14}4_{13}3_{10}$ (5)

3.4. Outline of HexNet routing scheme

The principal high level routing procedure of disaster information is as follows: 1) When a sensor node *senses* a physical change (potential disaster), it will respond to this by broadcasting an *alive request* to its neighboring nodes. Neighboring (1-hop) nodes that receive this message would then reply to the alive request stating their "alive" status. 2) Sensors *know* the number of neighbors in their vicinity (six in the normal case) and hence the number of replies they are expected to receive upon their alive request. 3) If a sensor node does not receive n_m replies, where n_m is the number of neighbors, it will issue a disaster message to its neighbors containing the RID_{AID} of nodes which have not replied, depending on a forwarding criterion (discussed in the next section). 4) Neighbors that receive this Alert will rebroadcast the message according to the forwarding criteria. This continues until the sink receives the disaster messages containing information about the destroyed nodes surrounding the disaster area. 5) The sink then sends the Alert messages to the appropriate authority centre via the Internet, where the region of disaster and disaster scale is determined using knowledge of the destroyed nodes, and the disaster is dealt with accordingly.

An example scenario of the above routing procedure was shown in Fig. 1. Several nodes fail (nodes 5_{15} 5_{16} and 6_{18}), and the neighboring nodes sense this, sending their alive request. However, the sensing nodes receive fewer than the expected number of replies. This is shown in Table 1. Each sensing node would then immediately discover which of its neighboring nodes has failed, and will issue the Alert message (according to the forwarding criteria discussed throughout the paper). The Alerts are routed to the sink (S), where they are then forwarded to the Disaster Management Centre via the Internet.

Table 2. Main node types and their function.

Node Type	**Function**
Failed/Destroyed Node	Nothing.
Sensing Node	1. Broadcast "alive request message" 2. Collect "alive" messages 3. Send Alert to neighbors*
Forwarding Node	Check the Alert message to see whether it is the best* node to forward. If so, broadcast, otherwise ignore.
Sink	Forward messages to Disaster Management Centre.

3.5. Node roles

Each node in the network that participates in sensing and routing has a status or type which will determine its function in the network. Table 2 shows the main roles of the nodes. In the table, procedures marked with an asterisk require additional criteria which will be described in the next section.

3.6. Alert forwarding scheme

There are several criteria for Alert message broadcasting and forwarding by nodes. These forwarding decisions are utilizable and are aimed at efficient use of node and network energy whilst retaining low transmission delay. A sensing node decides whether to send an Alert message after receiving *replies* from its neighbors containing information about their status (type) e.g. failed nodes, and their cost to the sink- which is a single mixed weighted metric based on the predicted hopcount to sink, current node energy level, and physical distance to sink.

Hop Count Predictor (HCP): A node can *predict* the number of hops to the sink from the knowledge of its failed neighboring nodes. The prediction is manifested and updated in a field called the *hop-count predictor* (HCP) in the Alert message header. This *integer* field is initially set to the Range ID of the current node. It will then increment the HCP depending on its next best potential node for forwarding, known as the *potential forwarding node* (PFN). The HCP is incremented according to the following rules, and further illustrated in Fig. 3. 1) Integer set to the Range ID initially. 2) Incremented by 1 if the PFN has a Range ID equal to the parent node. 3) Incremented by 3 if the next PFN has a Range ID greater than the parent node.

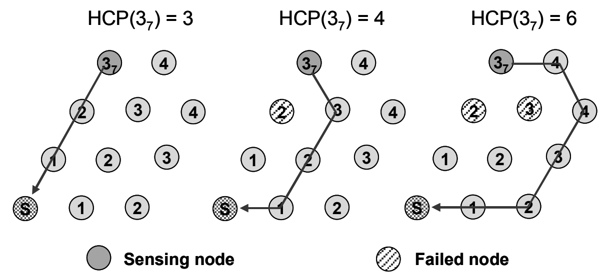

Figure 3. Example of HCP.

The HCP is used in order to make smarter decisions when forwarding Alert messages. Hence this is used as a parameter in the cost metric for decision making. Furthermore, the forwarding/broadcasting of Alert messages is based on *three* factors: 1) Current node's cost to sink. 2) Status (e.g. number of failed neighbors) and cost of neighboring nodes. 3) A maximum HCP and minimum energy required for a node to forward a message.

The cost of node X to the sink is calculated using the weighted sum of the energy and HCP and is given by

$$Cost_X = \frac{w_1}{E_X} + w_2 HCP_X + w_3 D_X \tag{4}$$

where E_X is the current *normalized* energy of node X, HCP_X is the HCP of node X to the sink, D is the physical distance of node X to the sink (calculated using RID_{AID}), w_1 and w_2 are the weights for the *significance* of (individual node) energy and delay in routing respectively, and $0 \le w_1, w_2 \le 1$. If $w_1 >> w_2$ then the energy factor in routing increases, and hence routes are primarily selected based on the energy of individual nodes in the network which will take part in routing decisions. Hence the routing of data might take a longer (hence more delay-inducing) path in order to save energy of individual nodes. However the consequence of this is that although the chance of an individual node running out of energy decreases, the chance of the total of number of nodes losing energy increases. Furthermore, $w_3 << w_1, w_2$ as D only serves as a simple discriminate when HCP and energy of two potential forwarding nodes are equal in value.

3.7. Local Neighbor Node Table (LNNT)

Each node posses a *local neighbor node table* (LNNT) containing information about its neighbor, such as their status, HCP, and the neighbors which are within a 1-hop range of each other. Table 3 shows a corresponding table which would be located at node 2_6. The LNNT can be used for routing Alerts and for decision making at each node. For example, in Fig. 4

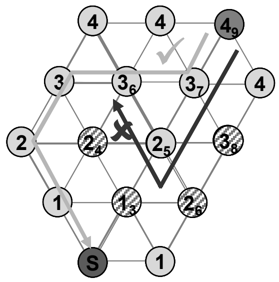

Figure 4. Routing using local neighbor information.

when node 2_5 receives an Alert from 4_9 via 3_7 it will *not* forward it, as it knows that 3_6 (being a neighbor of 3_7) can broadcast it with a smaller cost as it has a smaller HCP than it (assuming HCP is the only cost metric used in this case). Every node initially populates its table by setting the HCP field to the RID of its corresponding neighbors, the Status to ALIVE, and energy to the maximum node energy. These fields are then updated upon message exchanges with neighbors. These mechanisms are detailed in the following sections.

3.8. HexNet routing algorithm

The HexNet routing algorithm is as follows: 1) A sensor node *X* senses a physical change in its environment (heat/sound/electrical surge), which could be caused by a potential disaster. 2) Node *X* broadcasts an Alive Request (AREQ) to its 1-hop neighbors, and initiates a timer for Alive Reply (AREP) collection. 3) If node *X* also receives an AREQ, it will wait until it receives AREPs from its neighbors. 4) Once all the expected number of AREPs are received from *X*'s *alive* neighbors, or when the timeout is reached, node *X* will calculate its own HCP using knowledge of failed neighbors, then append this value to the AREP and broadcast it. At this time, the LNNT (Table 3) is updated. 5) Node *X* will then use the collected information (parameters shown in Table 3) to calculate the cost as in (4) and decide on Alert broadcasting.

It is important to note that although a sensing node does not receive an AREP from the neighboring sensor node at the same time as those from other alive nodes, it knows that the sensing node is *alive* as it had already received a AREQ from this node, hence it will wait until it receives the reply from this neighboring sensor within the timeout period. If all the expected AREPs are received before the timeout is up, then the Alert message is sent.

Table 3. Local neighbor node table (LNNT)

Node	Neighbors	Status	HCP	Energy
2_5	$1_3\ 2_4\ 3_6\ 3_7\ 3_8\ 2_6$	ALIVE	5	10
1_3	$2_4\ 2_5$	FAIL	N/A	N/A
2_4	$3_6\ 1_3$	FAIL	N/A	N/A
3_6	$2_4\ 3_7$	ALIVE	4	10
3_7	$3_6\ 3_8$	ALIVE	5	10
3_8	$3_7 2_6$	FAIL	N/A	N/A
2_6	$1_3\ 3_7$	FAIL	N/A	N/A

3.9. Packet format

There are three main packet types for the HexNet shown below.

Alive Request (AREQ) <RID_{AID}>
Alive Reply (AREP) <RID_{AID}><HCP><EL>
Alert Message
<RID_{AID}><FNNs (RID_{AID})><HCP><EL>

The FNN is the Failed Neighbor Nodes for which their RID_{AID} is provided in this field, and EL is the Energy Level of the current node with id of RID_{AID}. Once a node receives an Alert Message, it will make a decision based on the criteria given above. Each time an Alert Message is being forwarded, the three fields RID_{AID}, HCP and EL are updated with the corresponding values of the current node which is forwarding the message. We note that although the HCP in an AREP message is initially the same as the RID, it may become more accurate as the nodes learn more about the network environment and their failed neighboring nodes. Initially a sensing node that also receives AREQs (after sending its own AREQ) from neighboring sensor nodes will wait until it receives AREPs from its own neighbors before replying and broadcasting its own AREP. This is to ensure that the HCP that is forwarded in its AREP is updated from its local information. The HCP of neighbors is not used for calculating a node's own HCP, but used only for routing purposes. We also note that non-sensing nodes will initially reply with their HCP set their RID, as they have not "sensed" any failure in their neighborhood, and hence *assume* that they have a direct path to sink.

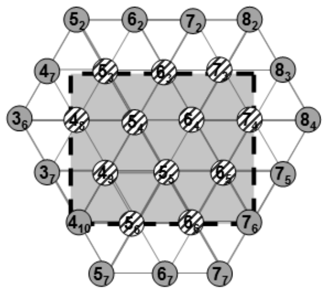

Figure 5. Disaster scale calculation.

3.10. Disaster-scale calculation

Once the Alerts are forwarded to the Disaster Management Centre, the "scale" of disaster can simply be calculated using the RID_{AID} of the failed nodes contained in the Alert messages to calculate the approximate area and position of the disaster. This final step becomes a trivial task, as the RID_{AID} of the failed nodes can simply be mapped back onto the HexMap, and presented on a graphical user interface (GUI) on the user's end. The (relative) accuracy of the disaster-scale calculation depends on three factors: 1) The physical detection (accuracy) of the sensor hardware. 2) Sensor node density. 3) The scale of disaster.

The physical detection may include the sensitivity of the detection mechanism of the sensor, and the required distance for detection. The sensor node density also affects the accuracy of the disaster scale size. In general, the higher the density of the sensor nodes (achieved by decreasing the physical distance between nodes), the higher the accuracy of the disaster-scale calculation at the sink. A simple rule is that the distance between nodes should be small enough for the most minor disaster (desired to be detected) to destroy nodes within an area of $d^2\sqrt{3}/4$, and the physical change to be detected or sensed within a distance of d by the neighbors of the destroyed nodes, assuming d is the distance between the nodes. Furthermore, a simple estimation of the disaster scale may involve taking the approximations of the horizontal and vertical distances of the disaster area using the following:

$$\max(D_i \cos\theta_i - D_j \cos\theta_j) \times \max(D_k \sin\theta_k - D_l \sin\theta_l) \quad (5)$$

where nodes i and j are the nodes which obtain the extreme horizontal distance of the disaster area, and nodes k and l are the nodes that obtain the extreme vertical distance of the disaster area. An example of this is shown in Fig. 5. In the figure, the (maximum) horizontal distance of the disaster is $3d$, whilst the (maximum) vertical distance of the disaster is $3\sqrt{3}d/2$. Hence the area of the disaster is estimated to be around $9\sqrt{3}d^2/2$. Hence the area of the disaster is estimated to be around $9d^2\sqrt{3}/2$.

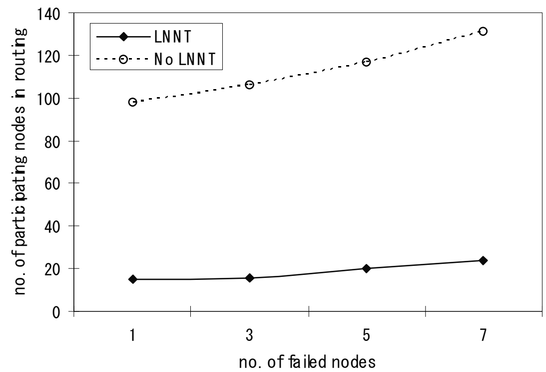

Figure 6. No. of participating nodes in routing Alerts.

4. Simulation of HexNet

The HexNet scheme is implemented and simulated in the Java platform. The scenario consists of 1260 nodes, spaced uniformly with a distance of 1.5 meters apart, across a field with a diameter of 20 nodes (60 m). The failed nodes are located at an angle of $\pi/3$ rads relative to the sink and a distance of RID = 15-17 hops. We set the cost weights w_1 w_2 w_3 to 1. In our model, each time a node broadcasts a message, the node loses *one* unit of energy. To demonstrate the effectiveness of the Local Neighbor Node Table (LNNT), two schemes are simulated: 1) HexNet Algorithm which does not use the LNNT. 2) HexNet Algorithm which uses the LNNT.

In the first scheme, a node will forward the Alert as long as its cost is lower than the parent node from which the Alert arrives from. However, in the second scheme, a node will only forward the Alert if it is the *best* node for forwarding the Alert. The node thus uses the LNNT table to evaluate itself as the best node. Fig. 6 shows the number of participating nodes in routing Alerts as the disaster-scale (number of failed nodes) increases for the two schemes. Fig. 7 shows the relative energy consumption of the network for the two schemes. In Fig. 7, it can be seen that the use of LNNT would significantly reduce the total energy used in the network, as it dramatically reduces the number of Alert rebroadcasts. This is due to the fact that only the lowest cost nodes in a neighborhood broadcast an Alert on the same failed node.

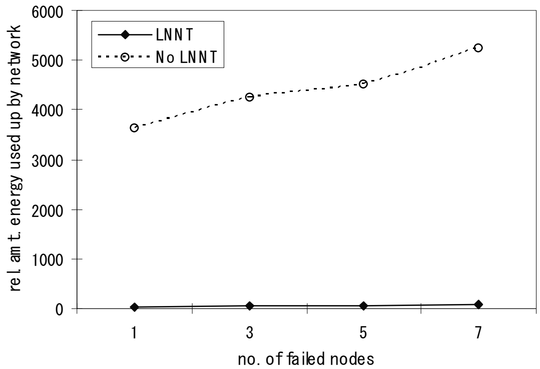

Figure 7. Rel. energy consump. in alert forwarding.

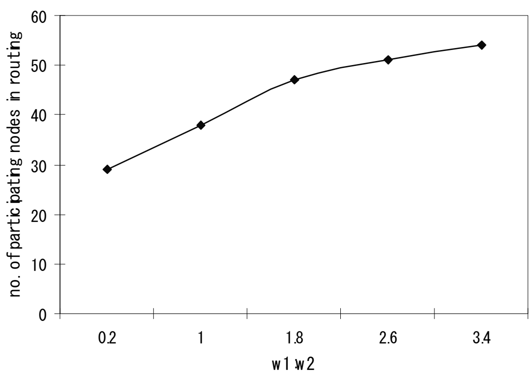

Figure 8. No. of participating nodes in alert routing.

In the case of not using a LNNT, every node that has a lower cost than the parent node from which the Alert was received from will rebroadcast the Alert. Furthermore, we investigate the effect of varying the ratio of cost parameters on the number of participating nodes in routing alerts, and the average energy used per node.

In the following simulation, the ratio of the energy to HCP $w_1 : w_2$ in the cost metric of (4) is increased. The number of failed nodes is set to 5, and the distance to failed nodes is set to RID = 15 hops. The physical distance metric weight, w_3 is set to 0.1. Fig. 8 shows the number of nodes that participate in routing alerts as this ratio is increased. From Fig. 8, it can be seen that as the ratio of energy to HCP increases, the number of nodes that participate in routing increase. This occurs, as the importance of saving individual node energies increases, hence nodes that have not already broadcasted an Alert would be given priority over their neighboring nodes that have already broadcasted or

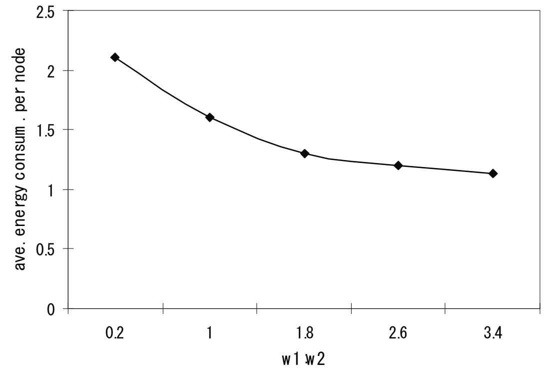

Figure 9. Ave. energy consump. per node.

forwarded an Alert, as they posses higher energy. Hence, a fresh new node is chosen over a node previously used, even though the older node may have had a lower HCP than the new node. The advantage of this is however reflected in Fig. 9. The figure demonstrates the reduction of average energy used by each participating node. The reduction is due to a node more willing to save its energy if it has already participated in Alert forwarding/broadcasting, giving this opportunity to its neighbors with higher energy levels.

5. Conclusion and future work

In this paper we introduce a new alert-based routing platform and mapping scheme for environmental field-based wireless sensor networks particularly for the reporting of disaster location and scale (area of disaster). Furthermore, we propose an efficient cost metric and routing scheme which can be used in the proposed platform. The proposed scheme dubbed HexNet is aimed at environmental sensor networks that are prone to destruction due to disasters and generally deprived of localization properties and capabilities such as limited or no GPS availability, and inherently limited energy source. Future work should aim at possibilities of reducing some of the strict requirements of the mapping scheme, further optimization of the parameters and weight selection, and finally additional performance evaluation of the proposed scheme.

6. Acknowledgement

This research was supported in part by the “Global COE (Centers of Excellence) Program” of the Ministry of Education, Culture, Sports, Science and Technology, Japan.

7. References

[1] J. Bachrach, and C. Taylor, “Localization in sensor networks,” in Stojmenović I., (Ed) *Handbook of Wireless Sensor Networks : Algorithms and Architectures*, pp. 277 – 310. New Jersey, Wiley Series on Parallel and Distributed Computing, 2005.

[2] P. Bahl and V. Padmanabhan, “RADAR: An in-building RF-based user location and tracking system,” in Proc. of IEEE INFOCOM’00, Mar. 2000.

[3] U. Bischoff, M. Stronbach, M. Hazas, and G. Kortuem, “Constraint-based distance estimation in ad-hoc wireless sensor networks,” in Proc. of EWSN’06, Zürich, Switzerland, Feb. 2006.

[4] C. Whitehouse, “The Design of Calamari : An ad-hoc localization system for sensor networks,” Master’s Thesis, University of California at Berkeley, 2002.

[5] A. Ward, A. Jones, and A. Hopper, “A new location technique for the active office,” IEEE Personal Communications, vol. 4, pp. 42-47. Oct. 1997.

[6] N. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller, “Anchor-free distributed localization in sensor networks,” in Proc. of SenSys’03, LA, USA, Nov. 2003.

[7] A. Savvides, C.-C Han,., and M. B Strivastava, “Dynamic fine-grained localization in ad-hoc networks of sensors,” in Proc. of MobiCom’01, Rome, Italy, Jul. 2001.

[8] D. Niculescu, B. Nath, “Ad hoc positioning system (APS) using AoA,” in Proc. of IEEE INFOCOM’03, Mar. 2003.

[9] R. Nagpal, H. Shrobe, J. Bachrach, “Organizing a global coordinate system from local information on an ad hoc sensor network,” in Proc. of IPSN’03, Palo Alto, USA, Apr. 2003.

[10] J. D. McLurkin, “Algorithms for Distributed Sensor Networks,” Master’s thesis, UCB, Dec. 1999.

[11] S. Guha, R. Murty, E. Sirer, “Sextant: a unified node and event localization framework using non-convex constraints,” in Proc. of MobiHoc’05, IL, USA, May 2005.

[12] G. Lukachan, M. Labrador and W. Moreno, “Scalable and energy-efficient routing for large-scale wireless sensor networks,” in Proc. of ICCDCS’06, Mexico, Apr. 2006.

[13] B. Karp, and H. Kung, “Greedy Perimeter Stateless Routing,” in Proc. of Mobicom 2000, Boston, USA, Aug. 2000.

[14] J. Chen, Y., Guan, U. Pooch,., "Customizing a geographical routing protocol for wireless sensor networks," in Proc. of ITCC 2005, vol.2, pp. 586-591 Vol. 2, Las Vegas, USA, Apr. 2005.

[15] J. Agre, and L. Clare, “An integrated architecture for cooperative sensing networks,” IEEE Computer Magazine, pp. 106-108, May 2000.

[16] M. Bhardwaj, T. Garnett, and A. P. Chandrakasan, “Upper bounds on the lifetime of sensor networks,” in Proc. IEEE ICC’01, Helsinki, Finland, Jun. 2001.

[17] P. Bonnet, J. Gehrke, P. Seshadri, “Querying the physical world,” IEEE Personal Communications, pp. 10-15., Oct. 2000.

[18] N. Bulusu, D. Estrin, L. Girod, and J. Heidemann, “Scalable coordination for wireless sensor networks: Self-configuring localization systems,” in Proc. of ISCTA’01, Ambleside, UK, Jul. 2001.

[19] W. R. Heinzelman, J. Kulik, H. Balakrishnan, “Adaptive protocols for Information dissemination in wireless sensor networks,” in Proc. of ACM Mobicom’99, pp. 174-185, Seattle, Washington, USA, Aug. 1999.

[20] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed diffusion: A scalable and robust communication paradigm for sensor networks,” in Proc. ACM Mobicom’00, pp. 56-67, Boston, MA, USA, Aug. 2000.

[21] C. Jaikaeo, C. Srisathapornphat, and C. Shen, “Diagnosis of sensor networks,” in Proc. IEEE ICC’01, Helsinki, Finland, Jun. 2001.

[22] J. M. Kahn, R. H. Katz, K. S. J., Pister, “Next century challenges: mobile networking for smart dust,” in Proc. of ACM Mobicom’99, pp. 271-278, Seattle, Washington, USA, Aug. 1999.

[23] S. Slijepcevic, and M. Potkonjak, “Power efficient organization of wireless sensor networks,” in Proc. of IEEE ICC’01, Helsinki, Finland, Jun. 2001.

[24] B. Warneke., B. Liebowitz, and K. S. J. Pister, “Smart Dust: communicating with a cubic-millimeter computer,” IEEE Computer Magazine, pp. 2-9, Jan. 2001.